International Journal of Computer Applications (0975 - 8887)
Volume 166 - No.8, May 2017

Power Optimization of CFD Applications on Heterogeneous Architectures

Maaz Ahmed
HKBK College of Engineering, VTU
Bangalore, Karnataka, India

Mohsin Khan
HKBK College of Engineering, VTU
Bangalore, Karnataka, India

Waseem Ahmed
King Abdulaziz University
Jeddah, KSA

Rashid Mehmood
King Abdulaziz University
Jeddah, KSA

Abdullah Algarni
King Abdulaziz University
Jeddah, KSA

Aiiad Albeshri
King Abdulaziz University
Jeddah, KSA

Iyad Katib
King Abdulaziz University
Jeddah, KSA

ABSTRACT

Computational fluid dynamics (CFD) is widely used by the scientific computing community to solve various fluid flow problems. High performance computing (HPC) enable faster CFD simulations with higher solution accuracies. Many CFD applications are run on multicore and multiprocessor platforms and are being increasingly ported to run on computational accelerators such as graphical processing units (GPUs) to increase the performance. The increase of computational power is necessary to allow faster computing, which in turn increases the power consumption. The issue of the power consumption has to be addressed, particularly in applications such as CFD, where the simulations may need to be executed iteratively for a large number of times, each iteration taking a large amount of time, in order to obtain appropriately accurate results. Power-aware computing is concerned with devising energy efficient methods. Techniques such as Dynamic voltage and frequency scaling (DVFS) and offlining can be used to reduce power consumption and increase the energy efficiency of applications. In this paper, we present a combination of these techniques and apply to CFD applications running on heterogeneous architectures. We reduce the overall power consumption with a small performance loss. Precisely, for DVFS, we save 4% to 23.5% of energy with a performance loss of 0.6% to 9.8%. Similarly, for online-offline mode, we save 22% of energy with a performance loss of 0.3%.

Keywords

Power-aware, DVFS, online-offline, Energy Efficiency, HPC, CFD

1. INTRODUCTION

The past decade has seen the growth of computational power of High performance super computers of up to petaflops. The emerging trends in technology may soon enable the use of exascale computing. The increase of computational power is necessary to allow faster computing, but this also increases the power consumption. The amount of power/energy required by these computing systems may not be accessible every time, due to infrastructure unavailability. Also the cost of running the hardware may outrun the cost of owning the hardware platform [1]. To address this issue the Scientific computing community are trying to build computer systems and applications that consume less energy.

Computational fluid dynamics (CFD) is widely used by the scientific computing community to solve various fluid (liquids and gases) flow problems. Many CFD applications are run on Multicore and multi-processor platforms and many CFD applications are being ported to run on the Accelerators to increase performance of the CFD applications. The infrastructure provided by the High performance computing systems enables the researchers to simulate the CFD applications much faster. Although the CFD applications are ported on the HPC systems with the aim of increasing the performance (execution time) of the application, the issue of (huge) power consumption is neglected. As mentioned previously, the issue of the power consumption has to be addressed, mainly in CFD applications since CFD applications are simulations that may require to run and re-run many times for a large amount of time (days to months) to obtain the appropriate and required accuracy levels. Hence there is a need of Power-aware computing when running CFD applications on Heterogeneous Computing Systems. The are many energy saving techniques that can be utilized to reduce the power consumption and increase the energy efficiency of the applications. Hence these techniques are combined in this research and applied to CFD applications that run on heterogeneous architectures and reduce the overall power consumption without a drastic change in the performance. We also compare the performance and energy efficiency of the applications with and without the energy saving techniques.

The main contributions of the paper can described as follows:

—We develop a combination of various techniques such as CPU online-offline, DVFS on CPU and GPU, idle wait (nanosleep) and C3 sleep states to increase the energy-efficiency on some of the heterogeneous CFD applications.

2. BACKGROUND

High Performance Computing (HPC) is the practice of aggregating computing power and parallel processing techniques for solving complex computational problems in the field of science, engineering, or business. In HPC, primary focus was based on performance, however as we are moving up from Terascale to Petascale

and approaching Exascale, power and energy have become critical concerns.

2.1 Power Consumption Sources

Power consumption can be broadly divided into dynamic power consumption and static power consumption. Static Power consumption arises due to Leakage Current and Reverse biased PN junction. The dynamic power consumption ($P_{dynamic}$) arises from charging and discharging of the load capacitance. The leakage power (P_{leak}) arises when device is inactive and still there is current flow.
Thus, we have

$$P_{dynamic} \propto CV^2f$$

$$P_{leak} = I_{leak}V$$

Where,
C = Load Capacitance
f = Operating Frequency
V = Operating Voltage
I_{leak} = Leakage Current

2.2 Importance of Power Management

Power management in HPC systems is extremely important but is challenging as it affects every part of a system. The cost of powering HPC systems has been steadily rising with growing performance, while the cost of hardware has remained relatively stable. If this situation continues to exist then energy cost of a large scale system could be more than the equipment itself during its life time. For example, Sunway TaihuLight consumes 15.371 MW of electricity. The cost to power and cool the system can be significant, i.e; 15.371 MW at $0.10/kWh is $1537 an hour or about $13.5 million per year.

2.2.1 Performance per Watt. HPC is used to deliver high performance, where complex problems which are compute intensive are solved using parallel processors. In Top500 list [2], speed decides the ranking of supercomputers. As the advancements in HPC is approaching towards Exascale computing, power and energy have become critical concerns. The Green500 list [3], ranks the supercomputer based on their power efficiency.

2.2.2 Power Challenges posed by exascale computing. Power is a major concern for all computing platforms. In HPC, high power results into high operational costs, effects environment due to high carbon emission, costs for cooling is increased and has an impact on reliability as the components may ware out very fast, which translates into lost productivity.

2.2.3 Green Computing. It has been estimated by the International Telecommunication Union (ITU) that the contribution of ICTs to environment is between 2% and 2.5% of total global carbon emissions. Hence Power management plays an important role in green computing.

3. METHODOLOGY

The solution to power saving and improving energy efficiency is to reduce the power usage of the system by altering the frequency and power profiles of the various components of the system. This must be done with limited performance loss (performance constraint) of the application. There are many areas which can be exploited to apply power saving techniques such as, when the application is being run on GPU with the CPU idle, when only one core of the CPU (serial portion of the application) is being executed and other cores are idle, by overlapping executions, by overlapping execution and communication, etc.
To reduce the power consumption and to increase the energy efficiency, various techniques such as DVFS, CPU online-offline etc are used in combination to achieve energy efficiency on CFD applications. Some of techniques that are applied in this paper are briefly described as follows:

3.1 DVFS

Dynamic Voltage and Frequency Scaling (DVFS) is the alteration of voltage and frequency levels of various processing elements in a system to optimize power saving. DVFS allows the processing elements to process the task using minimum power. Although DVFS may affect performance of the system it is majorly used for power optimization. In this research work the DVFS technique is applied on the CPU when serial portion of the application is being executed and when the application is being run on GPU with the CPU idle.

3.2 CPU online-offline

CPU online-offline is the logical switching on and off the CPU cores dynamically, that is, switching on and off the CPU cores when the system is up and running. CPU online-offline may also be referred to as hot-plug. The meaning of word logical here means that CPU online-offline only disables the work/task supply to CPU core but the power supply to the CPU core is still on. In this work the CPU online-offline is implemented on the CPU when the application is being run on GPU with the CPU idle and when only one core of the CPU is being used for running the application.

3.3 C-states

C-states are one of the power management techniques for CPU. They are idle power saving states and also known as sleep states. The higher the number of the C-state, the deeper the sleep state of the CPU, thus more power saving but incurs higher latency. C-states achieve idle power saving by progressive shutdown of the circuitry. C3 Sleep state is a state where all the core and bus clocks are turned off and also the clock generator is turned off. This is depicted in table 1.

3.4 Governors

In order to save power or to increase the performance of the system there are some tools available, some of which are built in the operating system. These tools can scale the frequency automatically based on the profile and system load. These tools have various profiles for frequency and power. In the Linux kernel there are various power schemes for CPU, which are known as governors. The governors and the description are defined in table 2. The ondemand and powersave governor also uses C-states to save power. Hence by using governors the technique of power saving by C-states is being utilized as well. In this research work the governors are applied when serial portion of the application is being executed and when the application is being run on GPU with the CPU idle.

Table 1. : C-states' Idle Power Level vs. Responsiveness

	C-State			
System	C0	C1	C3	C6/C7
Core Voltage	High	Medium	Medium	Very low
Core Clock	on	off	off	off
PLL	on	on	off	off
L1/L2 caches	saved	saved	flushed	flushed
LLC/L3 Cache	saved	saved	saved	saved/ flushed
Wake-up Time	Active	Small	Medium	Huge
Idle Power	Active	High	Medium	Low
Transition Energy	Active	Low	Medium	High

Table 2. : Governors in Linux OS

Governor	Description
ondemand	dynamic frequency scaling based on load. High frequency when high load and low frequency (C-3) states when low load.
performance	sets the CPU to maximum frequency.
powersave	sets the CPU to minimum frequency.
userspace	user specified frequencies are set for the CPU.

3.5 Idle wait

Idle wait (nanosleep) is a technique of switching the state of the processing elements to sleep state when they are in idle state. It can be triggered programmatically as well using the nanosleep function API. This state is activated during execution of the serial portion of the application program. When this state is activated the power saving is then taken care by the CPU governors.

4. EXPERIMENTAL TESTBED

The experiments were carried out on a High performance computing (heterogeneous) server with configurations as shown in table 3.

4.1 Server

INTEL XEON E5520: This processor is built on the Nehalem Micro-architecture. It is a two socket quad core processor. Each core has 2 threads which runs at 2.26 GHz and contains 32KB L1 instruction and data cache and has L2 cache of the size 256KB. The total size of the L3 cache is 8MB (shared). The size and type of the ram is 12GB and DDR3 respectively. This has a Thermal Design Power (TDP) of 80W.
NVIDIA QUADRO FX 3800: This graphics card has 192 cores, which has a frequency of 600MHz. It's memory size is 1GB and is of type GDDR3. It has a memory bandwidth of 51.2GBps. The PCI Express 2.0 x 16 interface is used to connect this graphics card with rest of the system. This has a TDP of 108W.

Table 3. : Configuration of the systems

	Server_1	
System	Intel Xeon E5520	NVIDIA QUADRO FX 3800
Micro architecture	Nehalem Micro architecture	Tesla Microrchi-tecture
Sockets	2	1
Cores per socket	4	192
Threads per socket	8	-
Core speed	2.26 GHz	600 MHz
Core size	45 nm	55 nm
Ram type	DDR3	GDDR3
Ram size	12 GB	1 GB
Memory bandwidth	25.6 GB/s	51.2 GB/s
L1 Data Cache	4 x 32K	-
L1 instruction Cache	4 x 32K	-
L2 Cache	4 x 256K	-
L3 Cache	8192K	-
OS	CentOS 7	-
Compiler	GCC 4.8.1 / nvcc	-
Power	80 W	108 W
Energy Efficiency	0.45 Gflops/W	5.77 Gflops/W

4.2 Power Measurement

Power measurement was done with a power meter with a frequency 1 Hz (1 sample per second). The power meter was attached to the power source on one side and the HPC main power source to the other side of the power meter, which is depicted in figure 1. Samples from the power meter were input to a separate system to avoid the probability of performance of the HPC system being affected.

5. RESULTS AND ANALYSIS

In this research work, the performance gains or losses has been explored on the basis of both power and energy savings. We applied different combinations of power management techniques on various CFD applications. These CFD applications have been taken from different sources - LUD and Euler were used from Rodinia

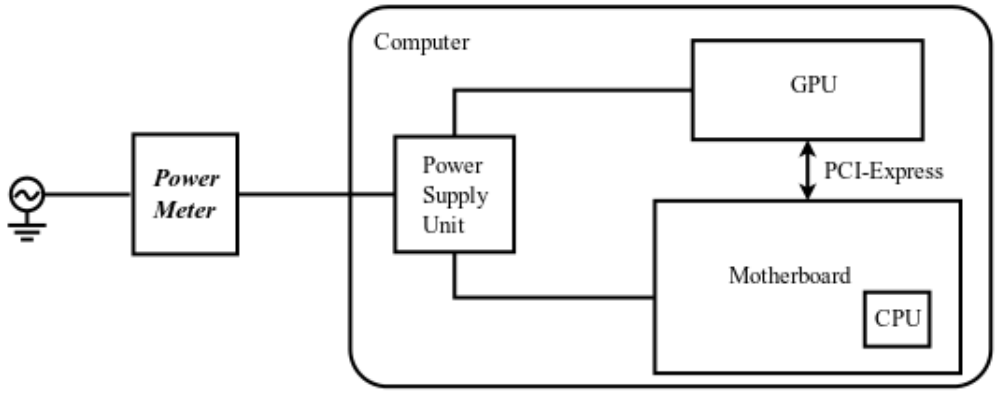

Fig. 1: Hardware System with Sampling Point

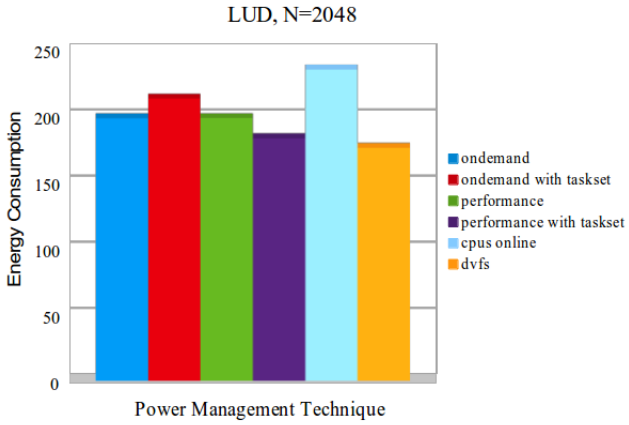

Fig. 2: Energy Consumption of LUD for N=2048

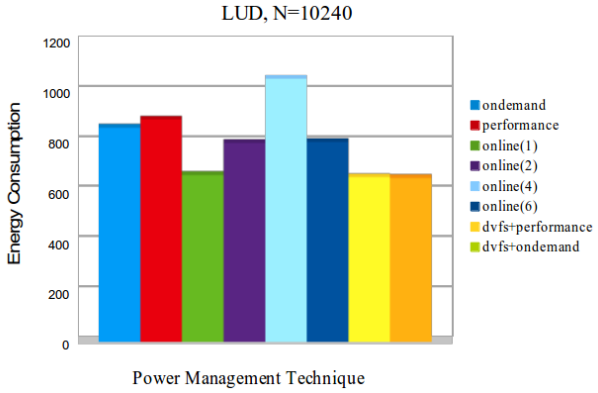

Fig. 3: Energy Consumption of LUD for N=10240

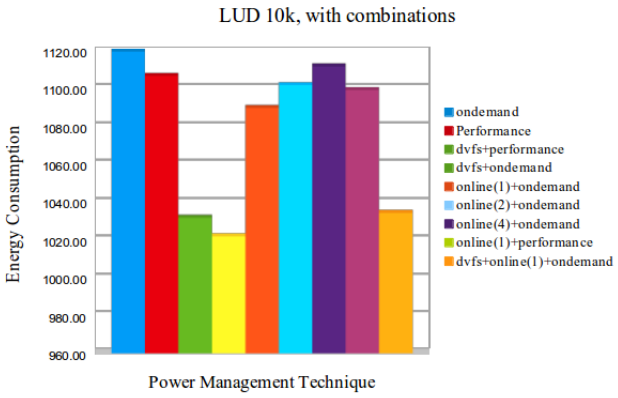

Fig. 4: LUD with different Power Management Combinations

Benchmark suite [4] and Lattice Boltzman Method (LBM) from Parboil Benchmark suite [5]. It was observed that LUD had the maximum energy savings with minimum performance loss. For LUD with matrix size as 2048, the energy saving was 11.27% and performance loss was 5% using DVFS. For LUD with matrix size of 10240, the energy saved and performance loss were 23.5% and 0.6% respectively using DVFS, 22% and 0.3% respectively using online-offline technique. These results are depicted in figures 2, 3, 4 and 5. For Euler using DVFS with input size as 97k, the energy saving was 4.1% and performance loss was 8%. and for input size of 193k, the energy saved and performance loss were 8.5% and 8.8% respectively. These are shown in figures 9 and 10. For LBM using DVFS with long input data set, the energy saving was 9.5% and performance loss was 9.8%. These results are depicted in figures 6, 7 and 8.

It was analyzed that as the input size (input data) increases we get better energy efficiency and a significant decrease in performance loss. It was observed that CPU online-offline technique gives better energy efficiency when compared to the default Linux governor (ondemand). It was also observed that the combination of DVFS with CPU online-offline was not as efficient as the DVFS technique.

6. RELATED WORK

The past few decades have seen various approaches to reduce power consumption in various fields. Low power and power-aware techniques to conserve energy have been used in Embedded Sys-

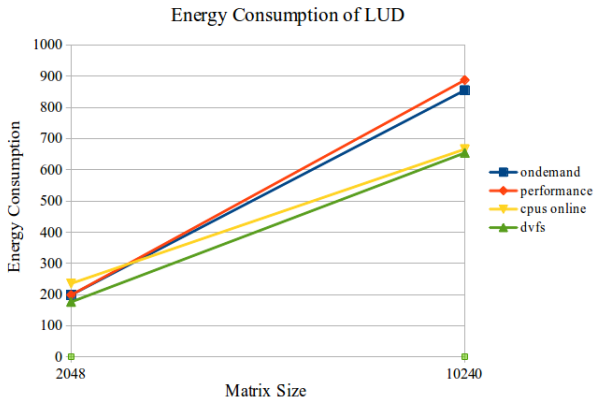

Fig. 5: Energy profile of LUD

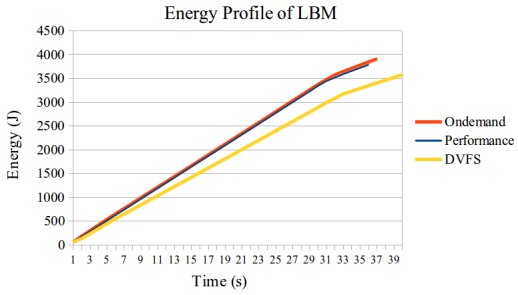

Fig. 6: Energy profile of LBM (Parboil)

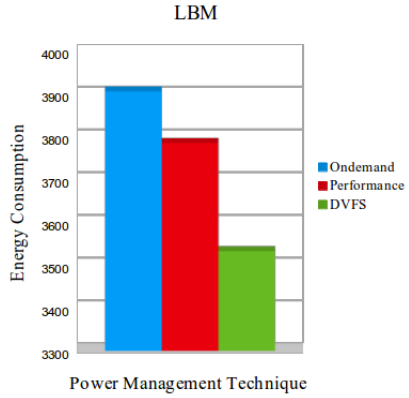

Fig. 7: LBM (Parboil) Energy Consumption

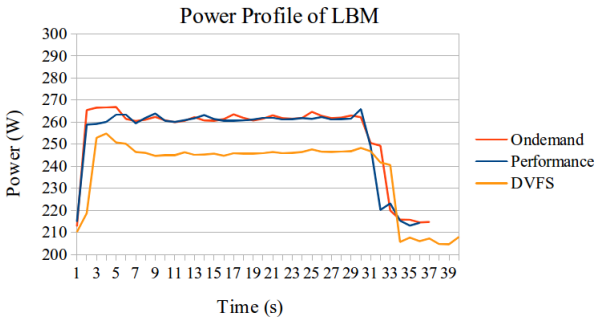

Fig. 8: Power profile of LBM (Parboil)

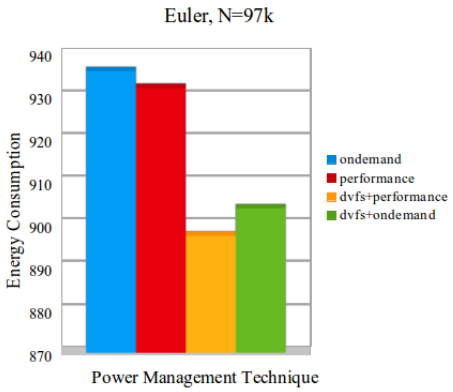

Fig. 9: Energy Consumption of Euler for N=97k

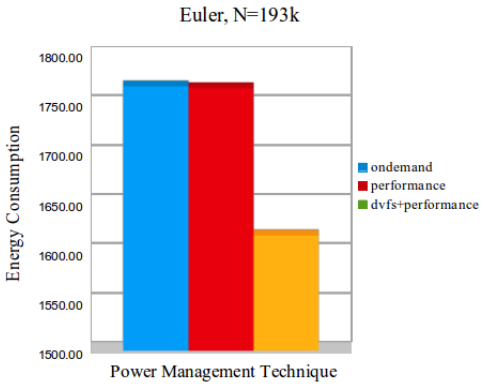

Fig. 10: Energy Consumption of Euler for N=193k

tems and Mobile Computing. The low power approach, which is effective in mobile and handheld systems, uses low power components to reduce power. However, in all these cases performance is limited. In contrast, the power- aware approach explores the trade-off between power consumption and performance attempting to find a best approach. Many of such approaches have migrated to all the major components of high performance systems including processor, disk, memory, and network card.

There are two main power management approaches a) Static Power Management (SPM) – Systems that utilize this technique use low power components to save energy, and b) Dynamic Power Management (DPM) - Systems that utilize this technique use software and power-scalable components to optimize energy consumption.

In static management, one of the popular approach is to replace high power components with their low power equivalents. For example, the Fast Array of Wimpy Nodes (FAWN) system [6][7], a novel cluster architecture for low power data intensive computing, combines low-power CPUs with small amounts of local flash storage. Energy saved from this architecture is two orders of magni-

tude more than disk based system, but there are some limitations in this architecture. This architecture may be effective for node scalable systems but ineffective in non-commodity clusters [8]. This also involves a major replacement of components which may not be possible in existing large infrastructures.

Recent studies use Dynamic Power Management for improving energy consumption. One of the approach is varying voltage independently using dynamic compilers with dynamic voltage scaling (DVS) techniques [9][10]. Dynamic Voltage and Frequency Scaling (DVFS) is another approach where both voltage and frequency are varied as required[11][12].

Ge et al. [13] presents a work where a framework proposed controls processor's frequency. This framework achieves a reduction of 36% with 5 % performance loss. In another work Freeh et al. [14] presents a work which uses techniques to reduce the processor's frequency in a cluster to reduce power consumption.

F. Alvarruiz et al. [15] proposed a work called CLUES which replaces idle state with power off state. In idle state energy savings of 3.42% were achieved, and 66.67% of energy was saved in power off state, it showed a energy saving of 66.67%.

We can see that, the states save power, but they directly effect the job execution time. No work so far has presented a comparison of all states to energy savings and its impacts on the jobs execution time.

7. CONCLUSION AND FUTURE WORK

As the demand for performance is increasing, there is an increase in demand for power efficient systems. If we can reduce power without affecting the performance we can increase the energy efficiency of the system. In this paper, we present an approach to improve the energy efficiency using combination of different power saving techniques for CFD applications. Dynamic Voltage and Frequency Scaling is used to scale the CPU frequency for the duration of the time the device is performing computations on GPU, this lowers the energy consumption. CPU online-offline technique is used to make the processor to go in the offline mode until the computations are completed on GPU. Our results show that for DVFS we save 4% to 23.5% of energy with performance loss of 0.6% to 9.8%. Similarly for offline mode we save 22% of energy with performance loss of 0.3%. Analyzing the results we can conclude that when we use combination of different power saving techniques in HPC systems with GPU acceleration, we can save more energy with limited performance loss. In future the power management technique, mainly DVFS, can be applied on GPU (similar to CPU) to get better energy efficiency for CFD applications on HPC systems.

8. REFERENCES

[1] S. Mittal, "A survey of techniques for improving energy efficiency in embedded computing systems," *International Journal of Computer Aided Engineering and Technology*, vol. 6, no. 4, pp. 440–459, 2014.

[2] "The top500 list," *http://www.top500.org (April 2017)*.

[3] "The green500 list," *http://www.green500.com (April 2017)*.

[4] S. Che, M. Boyer, J. Meng, D. Tarjan, J. W. Sheaffer, S.-H. Lee, and K. Skadron, "Rodinia: A benchmark suite for heterogeneous computing," in *Workload Characterization, 2009. IISWC 2009. IEEE International Symposium on*. Ieee, 2009, pp. 44–54.

[5] J. A. Stratton, C. Rodrigues, I.-J. Sung, N. Obeid, L.-W. Chang, N. Anssari, G. D. Liu, and W.-m. W. Hwu, "Parboil: A revised benchmark suite for scientific and commercial throughput computing," *Center for Reliable and High-Performance Computing*, vol. 127, 2012.

[6] D. G. Andersen, J. Franklin, M. Kaminsky, A. Phanishayee, L. Tan, and V. Vasudevan, "Fawn: A fast array of wimpy nodes," in *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*. ACM, 2009, pp. 1–14.

[7] V. Vasudevan, D. Andersen, M. Kaminsky, L. Tan, J. Franklin, and I. Moraru, "Energy-efficient cluster computing with fawn: Workloads and implications," in *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*. ACM, 2010, pp. 195–204.

[8] G. L. Valentini, W. Lassonde, S. U. Khan, N. Min-Allah, S. A. Madani, J. Li, L. Zhang, L. Wang, N. Ghani, J. Kolodziej *et al.*, "An overview of energy efficiency techniques in cluster computing systems," *Cluster Computing*, vol. 16, no. 1, pp. 3–15, 2013.

[9] O. Ozturk, M. Kandemir, and G. Chen, "Compiler-directed energy reduction using dynamic voltage scaling and voltage islands for embedded systems," *IEEE Transactions on Computers*, vol. 62, no. 2, pp. 268–278, 2013.

[10] Q. Shi, T. Chen, X. Liang, and J. Huang, "Dynamic compilation framework with dvs for reducing energy consumption in embedded processors," in *Embedded Software and Systems, 2008. ICESS'08. International Conference on*. IEEE, 2008, pp. 464–470.

[11] M. Etinski, J. Corbalán, J. Labarta, and M. Valero, "Understanding the future of energy-performance trade-off via dvfs in hpc environments," *Journal of Parallel and Distributed Computing*, vol. 72, no. 4, pp. 579–590, 2012.

[12] E. Le Sueur and G. Heiser, "Dynamic voltage and frequency scaling: The laws of diminishing returns," in *Proceedings of the 2010 international conference on Power aware computing and systems*, 2010, pp. 1–8.

[13] R. Ge, X. Feng, and K. W. Cameron, "Performance-constrained distributed dvs scheduling for scientific applications on power-aware clusters," in *Supercomputing, 2005. Proceedings of the ACM/IEEE SC 2005 Conference*. IEEE, 2005, pp. 34–34.

[14] V. W. Freeh, N. Kappiah, D. K. Lowenthal, and T. K. Bletsch, "Just-in-time dynamic voltage scaling: Exploiting inter-node slack to save energy in mpi programs," *Journal of Parallel and Distributed Computing*, vol. 68, no. 9, pp. 1175–1185, 2008.

[15] F. Alvarruiz, C. de Alfonso, M. Caballer, and V. Hern'ndez, "An energy manager for high performance computer clusters," in *Parallel and Distributed Processing with Applications (ISPA), 2012 IEEE 10th International Symposium on*. IEEE, 2012, pp. 231–238.